SYSTEMIC: Information System and Informatics Journal

ISSN: 2460-8092, 2548-6551 (e)

Vol 6 No 1 - Agustus 2020

Rancang Bangun *Service Application Program Interface Sistem Machine Learning* Klasifikasi Teks Menggunakan Algoritma *Support Vector Machine*

Ottoh Hidayatullah[1], Victor Amrizal[2], Arini[3]

[1,2,3)] Universitas Islam Negeri Syarif Hidayatullah

ottohhidayatullah@gmail.com[1], victor.amrizal@uinjkt.ac.id[2], arini@uinjkt.ac.id[3]

Kata Kunci

Support Vector Machine, Knowledge Acquisition, machine learning

Abstrak

Data menunjukkan angka yang sangat besar untuk penggunaan Internet di Indonesia. Bidang pendidikan, perpustakaan online adalah upaya memudahkan para peneliti untuk mencari referensi dokumen penelitian. Berdasarkan hasil observasi, UIN Jakarta sudah memiliki repositori dokumen penelitian yang baik, namun pada repositori dokumen penelitian online tersebut belum memenuhi fitur Knowledge Acquisition, kemampuan ini memungkinkan pengguna untuk memperoleh informasi pengetahuan yang tidak mudah langsung di dapat oleh pengguna. Pada penelitian ini dibangun sistem machine learning menggunakan algoritma Support Vector Machine untuk mengkategorikan dokumen berdasarkan bidang penelitian informatika. Penelitian ini juga membangun sistem services API (Application Program Interface) untuk digunakan oleh berbagai macam platform dan lingkungan sistem operasi yang berbeda. Akurasi dari sistem machine learning pada penelitian ini menghasilkan persentase akurasi klasifikasi 73,2% dengan memiliki nilai parameter 0,9. Pada tahap preprocessing pemilihan unigram-bigram adalah yang terbaik, tahap preprocessing yang menggunakan stemming mempengaruhi tingkat klasifikasi sistem machine learning namun mampu meningkatkan hasil akurasi kemampuan. Penggunaan jumlah data mempengaruhi akurasi kemampuan klasifikasi machine learning, terbukti dengan data ditambah menjadi 488 akurasi meningkat menjadi 74,49, dan data bertambah menjadi 492 data maka akurasi meningkat lagi menjadi 77,78%.

Keywords

Support Vector Machine, Knowledge Acquisition, machine learning

Abstract

Data shows very large numbers for Internet use in Indonesia. In the field of education, online libraries are an effort to facilitate researchers to search for references to research documents. Based on observations, UIN Jakarta already has a good repository of research documents, but the online research document repository does not fulfill the Knowledge Acquisition feature. This capability allows users to obtain knowledge information that is not easily accessible to users. This research build a machine learning system using the Support Vector Machine algorithm so that the system built can categorize documents based on the informatics research fields. This research also builds a system services API (Application Program Interface) so that data output from machine learning systems can be used by a variety of platforms and different operating system environments. The accuracy of the machine learning system in this study resulted in a percentage of classification accuracy of 73.2% with a parameter value of 0.9. At the preprocessing stage the selection of unigram-bigram is the best in this study. Preprocessing affects the level of classification of machine learning systems. Preprocessing using stemming improves the results of ability accuracy. The amount of data affects the accuracy of the machine learning classification ability, it can be seen when the data is increased to 488 accuracy increases to 74.49. When the experiment was done again so that the data increased to 492 data, the accuracy increased again to 77.78%.

1. Pendahuluan

Di bidang pendidikan khususnya di perguruan tinggi pemanfaatan informasi dapat digunakan sebagai referensi atau penunjang penelitian yang

DOI : 10.29080/systemic.v6i1.920

sedang dilakukan. Hampir seluruh perguruan tinggi memiliki perpustakaan online sebagai *repository* dokumen jurnal dan penelitian ilmiah. Kewajiban untuk memiliki perpustakaan *online* ini di dukung oleh kebijakan pemerintah Indonesia yang tercantum pada Undang - undang Nomor 43 Tahun 2007 Pasal 24 Ayat 3. Berdasarkan observasi pada *website repositoy Online* pada UIN Jakarta, sistem sudah memiliki kemampuan sebagai *Information Acces* dan *Text Organization*, terdapat fitur *Search Engine* pada Website *repository* perpustakaan *Online* UIN sehingga sistem mampu menyediakan informasi berdasarkan *input* berupa teks dari pengguna. Menurut [1], teks pada pengarsipan dokumen dapat memberikan informasi dengan baik jika memenuhi kemampuan *Information Acces, Knowledge Acquisition* dan *Text Organization.* Pengguna perpustakaan dengan mudah menemukan dokumen - dokumen yang tersimpan karena dokumen sudah dikelompokkan dengan susunan struktur yang baik dengan mengelompokkan dokumen tersebut berdasarkan Fakultas dan Jurusan yang ada di UIN Jakarta kemampuan ini sudah memenuhi kriteria *Text Organization*. Untuk kemampuan kriteria sistem manajemen dokumen berupa teks yang harus dimiliki berikutnya adalah kemampuan *Knowledge Acquisition*. Kondisi lain terkait pengguna yang mengakses internet menggunakan berbagai macam *platform* seperti *Web* app dan *Mobile* aplikasi, hal ini menuntut sistem yang mampu mampu terintegrasi dengan *platforms* yang terhubung internet. Untuk memfokuskan penelitian dalam melakukan pengkategorian dokumen ruang lingkup penelitian ini hanya menggunakan dokumen - dokumen jurusan Teknik Informatika sebagai bahan uji coba pembuatan sistem *machine learning* karena bidang tersebut sesuai dengan bidang keilmuwan peneliti.

Menurut [2], melakukan klasifikasi data berupa teks dalam bentuk berita menggunakan RapidMiner Studio ke dalam beberapa kategori, TFIDF untuk *feature extraction* dan *support vector machine* sebagai algoritma klasifikasi dengan menggunakan kernel trik RBF dan menghasilkan akurasi yang sangat baik.

Pada [3], melakukan klasifikasi teks Bahasa Indonesia, menggunakan teknik *crawling* dari website berita CNN Indonesia, membandingkan tingkat akurasi algoritma *support vector machine* dan *naïve bayes,* membandingkan pengaruh *feature extraction* SVD (*singular value decomposition)* dan TFIDF (*term frequency inverse document frequency*), hasil penelitiannya bahwa akurasi klasifikasi yang dihasilkan oleh *feature extraction* TFIDF dan algoritma *Mulinomial Naïve bayes* memberikan hasil yang terbaik.

Menurut [4], melakukan klasifikasi menggunakan *support vector machine* kernel *Polynomial* untuk data NUS SMS Corpus dengan jumlah data 40.000, dengan mengambil 6 kelas data, melakukan 2 percobaan yaitu klasifikasi dengan metode *one-vs-one* yaitu membedakan antara kelas satu dengan semua kelas yang ada yang menghasilkan akurasi 72,6 %., percobaan kedua menambahkan data pada setiap dilakukan proses permodelan, hasilnya adalah tingkat akurasi menghasilkan peningkatan akurasi pada tiap penambahan data.

Sedangkan pada [5], klasifikasi berita bahasa indonesia dengan membandingkan algoritma *support vector machine* dan *K-nearest Neighbo*, menggunakan *stopword* dan TFIDF sebagai *feature extraction*, hasil Algoritma *Support Vector Machine* lebih baik untuk melakukan klasifikasi teks di bandingkan dengan *K-Nearest Neighbor*. Jika dibandingkan antara dua jenis Kernel SVM, maka SVM *dengan Kernel Polynomial* menghasilkan akurasi yang lebih tinggi dengan SVM *Kernel Linier.*

Peneliti [6], klasifikasi berita dari media berita *online* Indonesia bahasa Indonesia, data berasal dari yaitu Kompas.com dengan 12 artikel kategori dengan tiap kategori diambil sebanyak 100 artikel (1200 data), membandingkan algoritma SVM dan NBC untuk mengetahui performa akurasi, *precision, recall*, dan *F-Measure*, dan pemprosesan dimana waktu SVM kernel RBF lebih baik dari NBC, peneliti menunjukkan ada 33 berita yang tidak dapat di prediksi dengan baik oleh kedua metode.

Pada [7], fitur *Chi-squared* memberikan hasil terbaik pada algoritma SVM dengan menggunakan metode *kernel,* menggunakan data 457 dokumen tumbuhan obat dan hortikultura yang berasal dari laboratorium temu kembali informasi IPB berbentuk XM. Jumlah data tersebut dibagi menjadi 70% data (320 dokumen) digunakan sebagai data *training* dan 30% (137 dokumen) dijadikan sebagai data uji. Nilai akurasi untuk *Kernel Linier, Polinomial,* dan *RBF* adalah 70.8%, 70.8%, dan 73.72%. Dari *feature selection Chi-square* menghasilkan akurasi 96.35%, 96.35%, dan 95.62%, sehingga pemilihan fitur chi-square membantu klasifikasi SVM dalam mengorganisasikan dokumen secara cepat, efisien, dan dapat meningkatkan kinerja sistem klasifikasi.

Berdasarkan dari studi literatur yang telah digunakan berikut adalah hal-hal yang menjadi fokus pada penelitian kami :

1. Mencari parameter terbaik pada algoritma *support vector machine* sehingga dapat meningkatkan tingkat akurasi klasifikasi
2. Mencari pengaruh proses *preprocessing stemming* dan *non-stemming* pada implementasi kinerja *support vector Machine* [8], [9].
3. Mencari pengaruh kinerja algoritma *support vector machine* dalam klasifikasi dengan menggunakan *unigram* dan *bigram* dengan bantuan metode *Chi-square* untuk melihat proses hasil fitur seleksi [10], [11]
4. Mencari pengaruh jumlah data pada *datasets* apakah mempengaruhi kinerja algoritma klasifikasi sistem *machine learning*
5. Merancang *services* API sehingga *machine*

learning dapat terintegrasi dengan *platform* yang berbeda [12].

6. Sistem *machine learning* ini akan menggunakan metode *support vector machine* dengan kernel trik dan metode *cross validation* sebagai optimasi akurasi.
7. Klasifikasi dokumen pada sistem *machine learning* diterapkan berdasarkan judul dokumen pada jurusan Teknik Informatika.
8. Sistem *machine learning* yang dibangun termasuk ke dalam *supervised learning.*
9. Sistem *Machine Learning* menggunakan data latih dan data uji dari dokumen skripsi repository UIN Jakarta jurusan Teknik Informatika.
10. *Modern Web App* dalam penelitian ini hanya digunakan sebagai bagian uji coba untuk menguji bagaimana *Service* yang dihasilkan dari Sistem *machine learning* dapat digunakan dan berjalan dengan baik [12].
11. Pengkategorian judul skripsi terbagi ke dalam beberapa bidang ilmu informatika yang terdiri dari *Artifical Inteligent, Data Science, Information Sercurity, Internet of Things, Network System, Software Engineer*. Pengkategorian dilakukan secara *manual* data tersebut dikategorikan berdasarkan teori yang penulis ambil dari buku dan jurnal.
12. Tools yang digunakan dalam membangung sistem *Machine Learning* klasifikasi dokumen menggunakan Visual Studio Code sebagai *IDE (Integrated Development Environtment)*
13. Menggunakan bahasa pemprograman Python dengan bantuan ScikitLearn sebagai Library [12], [13], dan [14].
14. Menggunakan MongoDb sebagai *Database* [15].
15. Angular Cli *Framework* untuk membangun *Modern Web App*
16. Menggunakan Docker sebagai teknologi integrasi.

2. Metode Penelitian

Dalam melakukan penelitian ini peneliti menggunakan :

1. Metode observasi dan studi pustaka untuk pengumpulan data.
2. Metode pengembangan sistem menggunakan *Continuous ML Model and Control Framework* [16], terdapat 4 fase yaitu *Plan & Acquire, Organize, Analyze Modeling, Deliver.*

2.1 Observasi

Peneliti melakukan proses *observasi content analisis* pada repositori *online* UIN Jakarta dengan domain http://repository.uinjkt.ac.id, untuk identifikasi struktur pola alamat website yang mengacu pada data dokumen skripsi jurusan teknik informatika menggunakan *library python* Scrapy [14].

Table 1. Identifikasi Website Repositori

No	Keterangan	URL
1	URL Halaman utama repositori UIN Jakarta	http://repository.uinjkt.ac.id/dspace/
2	URL repositori dokumen skripsi Teknik Informatika	handle/123456789/160
3	URL Parameter untuk mengurutkan data dari yang terbaru	Type=dataaccesssioned &sort_by=2& order=Desc&etal=7&submit_browse=Update
4	URL Parameter untuk mendapatkan jumlah data	rpp=480

Hasil Identifikasi URL : http://repository.uinjkt.ac.id/dspace/handle/123456789/160/browse?type=dateaccessioned&sort_by=2&order=DESC&etal=7&submit_browse=Update&rpp=480.

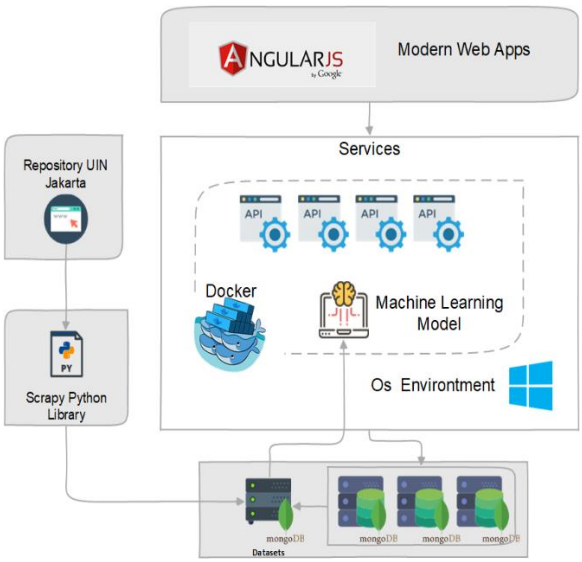
2.2 Continuous ML Model and Control Framework

2.2.1 Plan And Aquire

Pada tahap ini mengidentifikasi :

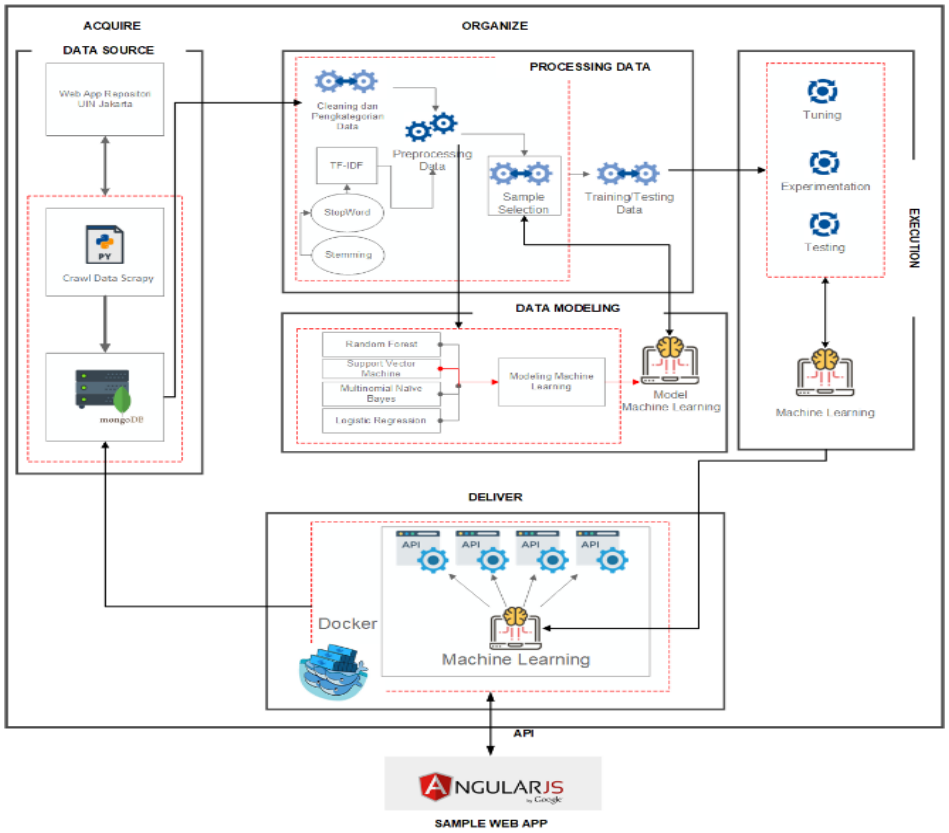
a. Tujuan dan sumber data.
Penelitian ini membangun *sistem Machine Learning* yang mampu melakukan klasifikasi teks dari dokumen skripsi sebagai data. Karena repositori UIN Jakarta belum memenuhi *knowledge aquistition*.

b. Arsitektur sistem dan *platform* pendukung yang digunakan. Gambar 1. adalah arsitektur desain sistem yang dibangun.



Gambar 1. Design Arsitektur Sistem

Perancangan sistem dapat dideskripsikan seperti pada gambar 2.



Gambar 2. Arsitektur End to end Machine Learning

2.2.2 Organize

Dari 489 data maka akan dilakukan pengkategorian dan pembersihan data, 80% sebagai data latih dan 20% untuk data uji. Selanjutnya menentukan *Category* sebagai *Class* fitur untuk klasifikasi data secara manual pada dataset. Terdapat 6 *Category* sebagai *Class* fitur klasifikasi data bidang informatika terdiri dari *Software Engineer (SE), Networking System (NS), Internet of Thing (IoT), Artifical Inteligent (Ai), Data Science (DS)* dan *Information Security (IS)*. Untuk mengetahui tingkat akurasi algorithma *Support Vector Machine* dibandingkan dengan algoritma lain seperti *Logistic Regression, Multinomial Naive Bayes, Random Forest Classifier* dengan bantuan *Library* Python SKLearn. Proses *organize* dibagi menjadi 2 tahap yaitu *feature engineering* dan *model engineering*.

2.2.2.1 Feature Engineering (Data Processing)

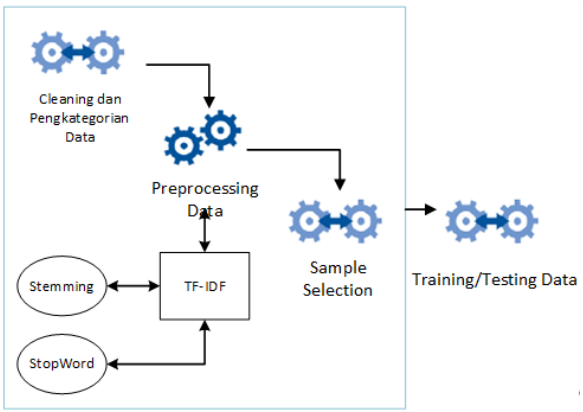
Tahapan ini melakukan pembersihan data (*cleaning*), pengkategorian data, dan *Preprocessing* yang meliputi Stemming, Stop Words, TF-IDF.

2.2.2.2 *Model Engineering (Data Modelling)*

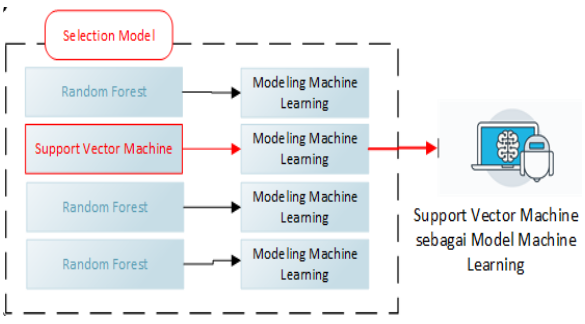
Tahap ini melakukan *model selection*, klasifikasi *Support Vector Machine* dan *Services API Machine Learning.*

1. Model Selection

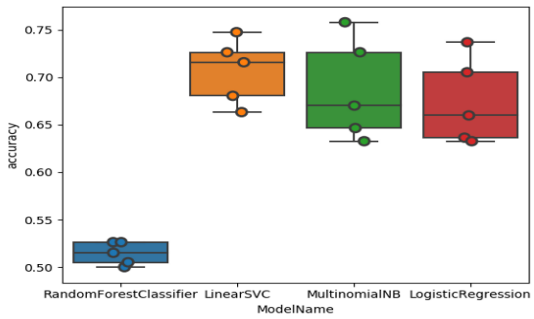
Pemilihan algoritma terbaik dilakukan untuk data yang sudah disiapkan pada proses *processing* dan *feature selection* yang bertujuan agar pada tahap selanjutnya sistem dapat fokus untuk mengoptimalkan kinerja sehingga menghasilkan model terbaik.



Gambar 3. Proses Tahapan Feature Engineering



Gambar 4. Proses Tahapan Model Engineering



Gambar 5. Hasil Klasifikasi Dengan Box Whisker Plot

Hasil direpresentasikan dalam diagram *whisker plot*, gambar 5. menunjukkan akurasi 65% hingga 75%, median data berada pada +/- 75% lebih tinggi dibandingan algoritma lainnya, dan akurasi total dari algoritma SVM mencapai 70%.

2. Klasifikasi Support Vector Machine

Klasifikasi menggunakan algoritma *support vector machine multiclass* dan menggunakan *kernel trick* untuk mentransformasikan data ke dalam dimensi *feature space* sehingga mampu mengurangi kesalahan klasifikasi data.

3. Services API Machine Learning

Berikut ini adalah API *services machine learning* yang disediakan pada sistem ini :

a) ***Insert Data,*** bertujuan agar judul skripsi dapat disimpan ke *database* agar jumlah *datasets* pada *database* terus bertambah sehingga *machine learning* dapat meningkatkan proses pembelajaran terhadap data baru. *Service* ini dapat diakses melalui *url /api/InsertData* dengan *method POST,* diuji menggunakan aplikasi *PostMan*, jika sistem mengembalikan nilai *boolean true* maka data berhasil disimpan, jika nilai *false* maka data gagal disimpan di *database.*

b) ***Check Data,*** bertujuan mengetahui distribusi data pada *dataset*. *Service* ini dapat diakses melalui *url /api/CheckData*, diuji dengan menggunakan aplikasi *PostMan*. *Method* yang digunakan adalah *GET* sehingga sistem langsung memberikan data dalam bentuk *array* jika ada *request* ke *url /api/CheckData* dengan *method GET.*

c) ***Train Model,*** bertujuan agar dapat melakukan pembelajaran pada *datasets* yang terus ditambahkan pada *service InsertData*. Data yang dibutuhkan dilakukan dengan melakukan input nilai parameter C, karena *kernel* trik linear membutuhkan parameter C. *Service* dapat diakses melalui *url /api/train* dengan *method POST.*

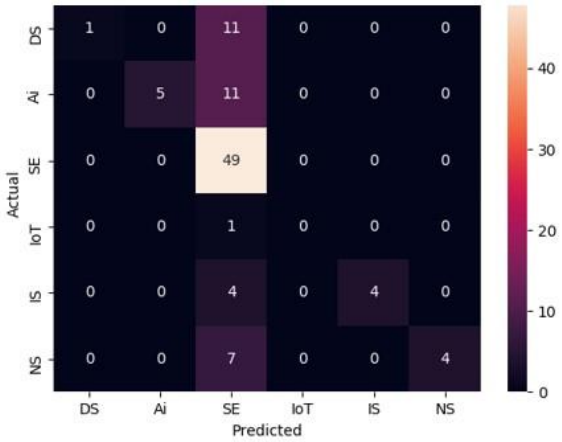
d) ***Build Classification,*** bertujuan untuk melakukan klasifikasi data yang di *input*. *Service Build Classification* dapat diakses melalui *url /api/BuildClassification* dengan *method POST*.

2.2.3 Analyze Modeling

Tahap ini melakukan evaluasi akurasi algoritma *Support Vector,* yaitu dengan melakukan *Experimentation*, *testing*, dan *tuning*.

1. Model Evaluation

Dari model *machine learning* yang telah dibuat maka dilakukan perhitungan tingkat akurasi. Perhitungan tingkat akurasi ini dilakukan dengan cara model melakukan klasifikasi data uji dengan data latih sebagai referensi. Untuk menghitung tingkat akurasi dengan menggunakan nilai *confusion matrik*, dibuat menggunakan bahasa pemprograman Python.



Gambar 6. Hasil Confusion Matriks

Tabel 2. adalah hasil perhitungan *atribute confusion matrix* :

Table 2. Atribut Confusion Matrik klasifikasi Support Vector Machine

No	Category Data	TP	FP	FN	TN	Jumlah Data
1	DS	1	0	11	85	97
2	Ai	5	0	11	81	
3	SE	49	34	0	14	
4	IoT	0	0	1	96	
5	IS	4	0	4	89	
6	NS	4	0	7	86	

Selanjutnya menghitung nilai *precision, recall, f1-score support* dari setiap kelas.

	precision	recall	f1-score	support
DS	1.00	0.08	0.15	12
Ai	1.00	0.31	0.48	16
SE	0.59	1.00	0.74	49
IoT	0.00	0.00	0.00	1
IS	1.00	0.50	0.67	8
NS	1.00	0.36	0.53	11
avg / total	0.78	0.65	0.59	97

Akurasi Model 64.95 %

Gambar 7. Hasil Nilai precision, Recall, F1-score, Support dan Akurasi

Hasil menunjukkan tingkat akurasi 64,95 %. Sistem mampu mengatasi *error* klasifikasi dilihat dari tinggi nya nilai *precision* pada tiap kelas dan total *precision* cukup baik yaitu 78% namun sistem memiliki tingkat yang sangat buruk pada kelas IoT, karena jumlah data yang sangat sedikit hanya berjumlah 7 data dari total data. Persentase *recall* pada tiap kelasnya belum memiliki nilai yang baik karena hanya kelas SE dan IS yang terbaik di antara lainnya, SE memiliki persentase 100% sedangkan IS memiliki persentase 50%, total persentase keseluruhan *recall* adalah 65%, dapat disimpulkan bahwa kemampuan klasifikasi *machine learning* yang dibangun sudah baik karena sudah melewati angka 50% hanya saja kemampuan klasifikasi machine learning masih terbilang kurang baik jika dilihat pada setiap kelasnya.

2. Performance Engineering

Tahapan ini untuk melakukan peningkatan kemampuan model *machine learning*, dilakukan dengan proses : *Tuni*ng, *Experimentation*, dan *Testing*.

a. Tuning

Tuning adalah proses pencarian atribut *terbaik* pada suatu algoritma. Pada percobaan pencarian atribut terbaik, menggunakan perbandingan *kernel trik linear* dan *RBF (radial basis function. Kernel linear* membutuhkan atribut C (konstanta = K), RBF membutuhkan atribut C (konstanta = K) dan γ (gamma). Penelitian ini menggunakan *function GridSearchCV* untuk membagi proses kerja kedua kernel, melakukan *cross validation* yaitu membagi *dataset* untuk data latih menjadi beberapa bagian dan pengujian dilakukan berulang kali sebanyak pembagian data latih yang dilakukan.

```
LinearSVC(C=0.9, class_weight=None, dual=True, fit_intercept=True,
    intercept_scaling=1, loss='squared_hinge', max_iter=1000,
    multi_class='ovr', penalty='l2', random_state=None, tol=0.0001,
    verbose=0)
          precision    recall  f1-score   support

       DS       0.58      0.50      0.54       14
       Ai       0.70      0.47      0.56       15
       SE       0.81      0.88      0.84       52
      IoT       1.00      1.00      1.00        2
       IS       0.38      0.50      0.43        6
       NS       0.75      0.75      0.75        8

avg / total       0.73      0.73      0.73       97

Akurasi Model 73.2 %
 ----- The SVM model has finished training in 1.0772520375240473 second ----- !
The Script Executed 1.0775012969970703 second !
PS C:\Users\ottoh\LearnML\Skripsi Machine Learning - Text Classification Repository UIN JKT\ProjectClassific>
```

Gambar 8. Hasil Akurasi Model Setelah Tuning

Peneliti mendefinisikan jumlah *cv* adalah 5, sehingga menghasilkan data latih sebanyak 97 data. Untuk *kernel* RBF peneliti mendefinisikan C dan γ (gamma) dari 1×10^5 hingga 1×10^{-5}, dan *kernel linear* mendefinisikan C dari 1×10^3, 0.9, 0.7, 0.4, 0.3, 1×10^{-1} hingga 1×10^{-5}. Dari hasil perhitungan kernel linear menghasilkan kernel terbaik untuk digunakan dengan nilai rata–rata 0.705 dan nilai std +/- 0.072 dengan nilai parameter C = 0.9.

b. Experimentation

Peneliti melakukan beberapa hal yang dianggap mempengaruhi tingkat akurasi klasifikasi model *machine learning* dengan memasukkan nilai pembobotan dimana pembobotan tersebut adalah hasil kalkulasi yang dilakukan terhadap hasil *processing* untuk itu peneliti melakukan percobaan pada metode *stemming* pada *preprocessing*. Peneliti melakukan pengujian pembobotan dengan menggunakan *bigram-trigram, trigram-quadgram* dan *unigram-bigram* dan bagaimana akurasi model jika data ditambah menjadi 492 data.

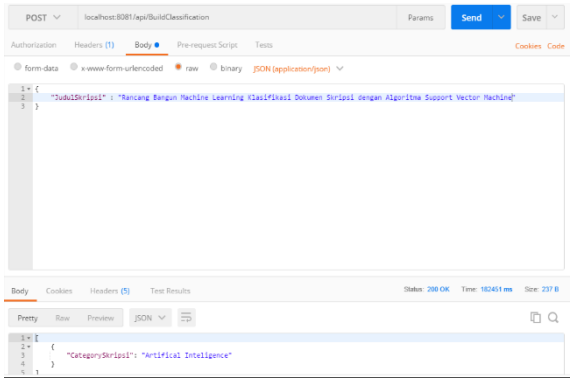
```
PROBLEMS 8    OUTPUT    DEBUG CONSOLE    TERMINAL

Jumlah Dataset :  492
Jumlah Data Skripsi Title : 490
Jumlah Data Skripsi Abstrak : 105
Jumlah Data Category :
 SE     241
Ai      74
NS      63
DS      55
IS      48
IoT     11
Name: SkripsiCategory, dtype: int64
[{'SE': 241}, {'Ai': 74}, {'NS': 63}, {'DS': 55}, {'IS': 48}, {'IoT': 11}]
-------------Specification Model ! -------------
 LinearSVC(C=0.9, class_weight=None, dual=True, fit_intercept=True,
     intercept_scaling=1, loss='squared_hinge', max_iter=1000,
     multi_class='ovr', penalty='l2', random_state=0, tol=0.0001,
     verbose=0)
          precision    recall  f1-score   support

       DS       0.43      0.67      0.52         9
       Ai       0.82      0.69      0.75        13
       SE       0.85      0.83      0.84        54
      IoT       1.00      0.80      0.89         5
       IS       0.62      0.83      0.71         6
       NS       0.89      0.67      0.76        12

avg / total       0.81      0.78      0.79        99

Akurasi Model 77.78 %
 ----- The SVM model has finished training in 2.9999999640040187e-07 second ----- !
The Script Executed 59.72904992103577 second !
PS C:\Users\ottoh\LearnML\Skripsi Machine Learning - Text Classification Repository UIN JKT\ProjectClassific>
```

Gambar 9. Hasil Akurasi Model Eksperimen

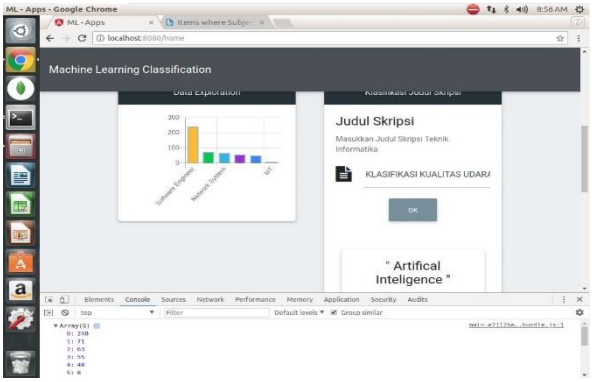
Dari hasil eksperimen diketahui bahwa *preprocessing* dengan menambahkan proses *stemming* sangat berpengaruh, yaitu dari persentase akurasi 73,2% menjadi 77,32% serta nilai *precision, recall,* dan *f1-score* memiliki nilai yang seimbang. Hal ini menunjukkan bahwa model *machine learning* mampu menghindari *error* klasifikasi yang baik sebanding dengan kemampuan model *machine learning* untuk melakukan klasifikasi. Akan tetapi proses pembobotan yang melibatkan pemilihan kata menggunakan *bigram-trigram* dan *trigram-quadgram* memberikan dampak pada penurunan akurasi model *machine learning*. Jumlah data mempengaruhi akurasi kemampuan klasifikasi *machine learning*, hal ini ditunjukkan pada pengujian dengan data 488 akurasi meningkat menjadi 74,49 menggunakan kernel *rbf*, nilai C = 1×10^5, dan nilai $\gamma = 1 \times 10^5$ (gamma). Ketika dilakukan percobaan dengan data bertambah menjadi 492 data maka akurasi meningkat menjadi 77,78% dengan kernel *linear* dan parameter C = 0,9.

c. Testing

Tahap ini menguji akses *services* API menggunakan aplikasi *Postman* terhadap aplikasi berbasis web yang telah dibuat pada tahapan *Model Engineering*.



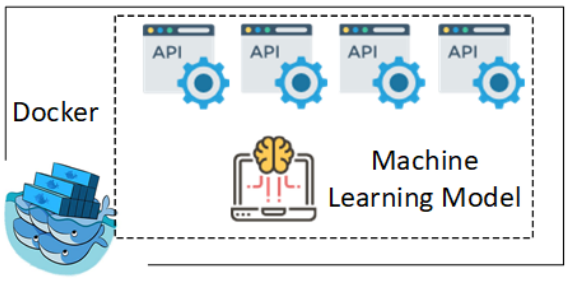
Gambar 10. Testing Output BuildClassification Service API Menggunakan Postman



Gambar12. Service API CheckData dan BuildClassification Oleh Web App

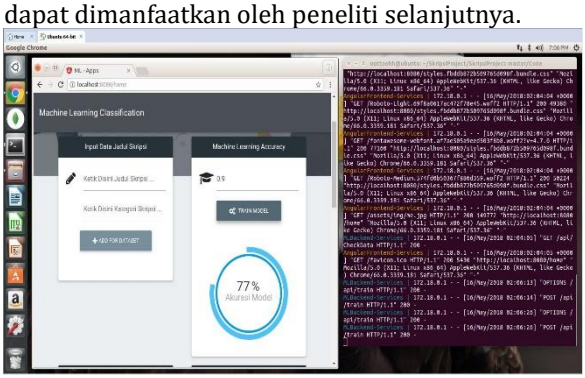
2.2.4 Deliver (Deployment)

Tahapan ini bertujuan agar sistem *machine learning* dapat menjadi *services* API (*aplication program interface*).



Gambar 11. Proses Tahapan Deployment

Dengan menggunakan *Flask Python* untuk membangun *services API machine learning* dan akan menambahkan kemampuan agar dapat berjalan diatas sistem operasi yang berbeda dengan menggunakan *Docker* sehingga sistem *services* API dapat dimanfaatkan oleh pengguna yang memiliki sistem operasi yang berbeda, juga dapat dimanfaatkan oleh peneliti selanjutnya.



Gambar 11. Tampilan Interface Services Sistem MachineLlearning dan Modern Web Apps Yang Sudah Terintegrasi.

Pengujian yang dilakukan dengan berbasis web ini yaitu menggunakan aplikasi berbasis web yang memiliki konsep *modern web apps* dengan menggunakan *framework* AngularJs.

3. Hasil Dan Pembahasan

Penelitian ini telah berhasil melakukan pengumpulan data, dan melakukan perencanaan (*Plan and Aquire)* dengan melakukan desain arsitektur secara umum dan arsitektur tahapan *End to End ML Architecture*) pada setiap prosesnya, tergambar pada gambar 2.

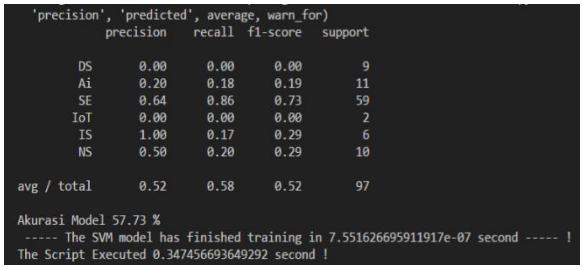
Selanjutnya pada tahapan *Organize*, telah melakukan persiapan data, data dapat dijadikan obyek klasifikasi pada algoritma SVM, yaitu proses *feature engineering* dan *model engineering*. *Feature engineering* berfungsi untuk melakukan persiapan data mulai dari proses pembersihan data, pengkategorian data, *stemming*, *stopword,* dan pembobotan menggunakan TFIDF (*term frequency inverse document frequency*) sehingga menghasilkan data yang mampu dibaca oleh algoritma yang digunakan. Pada *Model engineering* berfungsi melakukan proses pengujian algoritma *support vector machine* dengan algoritma lain untuk proses klasifikasi yaitu *logistic regression, random forest, multinomial naïve bayes*, yang menghasilkan bahwa algoritma *support vector machine* berada pada akurasi yang baik, terlihat pada gambar 5.

Pada tahapan *Analyze* perancangan telah dapat digunakan untuk mengetahui kemampuan klasifikasi dengan algoritma *support vector machine*. Hal tersebut dilakukan dengan m*odel evaluation,* telah diketahui seberapa baik sistem *machine learning* menghindari tingkat kesalahan klasifikasi (*precision*), seberapa baik sistem *machine learning* mampu melakukan klasifikasi (*recall*) dan porsi kemampuan keduanya dengan Kemampuan *f1-score*, dengan hasil nilai akurasi *precision* sebesar 78%, *recall* sebesar 65%, *f1-score* sebesar 59%, dan total akurasi sebesar 64,95%. Nilai-nilai tersebut menunjukkan hasil yang baik, terlihat pada gambar 7.

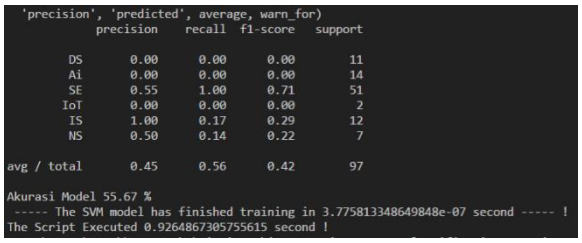
Upaya untuk meningkatkan tingkat akurasi dilakukan pada tahapan *performance engineering* dengan menggunakan 2 proses yaitu *tuning* dan *Experimentation*. Dengan *Tuning* telah dapat melakukan proses pencarian metode dan parameter terbaik yang mampu meningkatkan akurasi model *machine learning*. Pencarian

parameter terbaik menggunakan metode *cross validation* dengan nilai 5, sehingga data latih dibagi menjadi 5 bagian dan setiap bagiannya digunakan oleh model untuk dilakukan proses pengujian terhadap data uji. Hasilnya ditemukan metode terbaik yaitu menggunakan *kernel linear* dengan parameter C yaitu 0,9 dengan metode dan parameter tersebut tingkat akurasi klasifikasi *machine learning* meningkat menjadi 73,2 %. Nilai ini menunjukkan telah ada peningkatan *performance,* terlihat pada gambar 8.

Pada proses *Experimentation* telah mencari kemungkinan yang mampu meningkatkan kemampuan akurasi sistem *machine learning*, yaitu aspek pengaruh proses *stemming* pada *preprocessing* data, dan pengaruh pemilihan n-gram. Pengaruh *stemming* pada persiapan data ditemukan bahwa dengan menggunakan *stemming* akurasi dapat meningkat menjadi *precision* 78%, *recall* 77%, *f1-score* 77%, dan total akurasi menjadi 77,32 % walaupun pemprosesan program berjalan menjadi sangat lambat ketika menggunakan stemming, terlihat pada gambar 9.



Gambar 13. Akurasi Bigram-Trigram

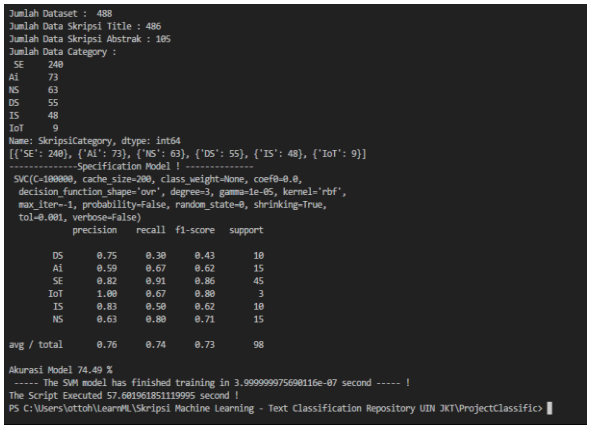


Gambar 14. Akurasi Trigram-Quadgram

Pada tahapan *model evaluation* proses *preprocessing* menggunakan *unigram* dan *bigram*, kemudian peneliti melakukan eksperimen terhadap pemilihan n-*gram*, hasilnya adalah ketika menggunakan *bigram-trigram* nilai total akurasi menurun menjadi 57,73 % dan ketika menggunakan *trigram-quadgram* nilai total akurasi mengalami penurunan lagi sehingga total akurasi menjadi 55,67%, terlihat pada gambar 13 dan gambar 14.

Terkait dengan pengaruh jumlah data yang digunakan, terbukti bahwa jumlah data mempengaruhi nilai akurasi kemampuan klasifikasi *machine learning*, terlihat pada data ditambah menjadi 488 akurasi meningkat menjadi 74,49 %, terlihat pada gambar 15.

menggunakan kernel *rbf*, nilai C = 1×10^5, dan nilai $\gamma = 1 \times 10^5$ (gamma). Ketika dilakukan percobaan kembali sehingga data bertambah menjadi 492 data maka akurasi meningkat lagi menjadi 77,78% dengan kernel *linear* dan parameter C = 0,9, ditunjukkan pada gambar 9.



Gambar 15. Akurasi Penambahan Data 488

Pada tahap *deliver menunjukkan bahwa* sistem telah dapat terintegrasi dengan *platform* serta dapat berjalan pada lingkungan (*environtment*) sistem operasi yang berbeda yang menggunakan *library Flask Python*. Data *services* API disajikan dalam format JSON (*javascript object notation*) dengan tujuan agar data dapat diakses melalui HTTP protokol yang sudah tersedia dan dapat diakses oleh berbagai macam jenis *platform*, selain itu juga menggunakan *modern web apps* untuk mengimplementasikan *services* API yang dapat digunakan oleh aplikasi berbasis web. Untuk mendukung sistem yang mampu berjalan di lingkungan (*environtment*) sistem operasi yang berbeda menggunakan teknologi Docker, ditunjukkkan pada gambar 11 dan gambar 12.

4. Kesimpulan

Penelitian ini berhasil merancang *services* API sistem *machine learning* klasifikasi teks judul skripsi sesuai dengan desain arsitektur sistem yang dirancang pada tahapan pengembangan sistem. Metode dan parameter terbaik adalah algoritma *support vector machine* dengan parameter nilai 0.9, yang menghasilkan persentase akurasi klasifikasi 73,2%.

Pada tahap *preprocessing* pemilihan *unigram-bigram* adalah yang terbaik pada penelitian ini, hal ini dibuktikan ketika menggunakan *bigram-trigram,* dan *trigramquadgram* tingkat akurasi klasifikasi sistem *machine learning* mengalami penurunan. *Preprocessing* data berpengaruh pada tingkat klasifikasi sistem *machine learning*. *Preprocessing* menggunakan *stemming* meningkatkan hasil akurasi kemampuan klasifikasi sistem *machine learning* namun

menghabiskan waktu sekitar 2 menit untuk melakukan pemprosesan program.

Jumlah data juga mempengaruhi akurasi kemampuan klasifikasi *machine learning*, dapat terlihat ketika data ditambah menjadi 488 akurasi meningkat menjadi 74,49, Ketika dilakukan percobaan kembali sehingga data bertambah menjadi 492 data maka akurasi meningkat lagi menjadi 77,78%.

Pada penelitian ini berhasil mengintegrasikan *services* API *machine learning* pada *platform* yang berbeda dibuktikan dengan melakukan uji coba aplikasi berbasis web dengan memanfaatkan *services* API yang sudah dibangun, juga berhasil menerapkan kemampuan pada sistem *machine learning* klasifikasi dapat berjalan pada lingkungan (*environtment*) sistem operasi yang berbeda menggunakan teknologi Docker, dibuktikan dengan sistem dapat berjalan dengan mudah pada sistem operasi Ubuntu Linux.

Daftar Pustaka

[1] C. X. Zhai, and S. Massung, "Text Data Management and Analysis: A Practical Introduction to Information Retrieval and Text Mining." Penerbit New York : Association for Computing Machinery and Morgan & Claypool , New York. Vol 1, p.7-9, 2016.

[2] S. M. H. Dadgar, M. S. Araghi , and M. M. Farahani, " A Novel Text Mining Approach Based on TF-IDF and Support Vector Machine for News Classification," IEEE International Conference on Engineering and Technology (ICETECH), 2016, DOI: 10.1109/ICETECH.2016.7569223

[3] R. Wongso, " *News Article Text Classification in Indonesian Language*,", Procedia Computer Science 116:137-143 · December 2017, DOI: 10.1016/j.procs.2017.10.039

[4] M. Kretchmar , and Y. Zhao, *"Text Message Authorship Classification Using kernel Support Vector Machines," IEEE* International Conference on Computational Science and Computational Intelligence, 2014, DOI: 10.1109/CSCI.2014.121

[5] S. N. Asiyah, and K. Fithriasari , *Klasifikasi Berita online menggunakan Support Vector Machine dan K-nearest Neighbor, Jurnal Sains dan Seni ITS, Vol 5, No 2 (2016) DOI: 10.12962/j23373520.v5i2.16643*, http://ejurnal.its.ac.id/index.php/sains_seni/article/view/16643

[6] *D. Ariadi, and K. Fithriasari, "*Klasifikasi Berita Indonesia menggunakan *Naive Bayesian* dan *Support Vector Machine* dengan Confix Stripping Stemmer," *Jurnal Sains dan Seni ITS, Vol 4, No 2, 2015,* DOI: 10.12962/j23373520.v4i2.10966

[7] A. D. Putri, and J. Adisantoso, " Klasifikasi dokumen teks menggunakan metode *Support Vector Machine* dengan pemilihan fitur *CHI-Square*," 2013, http://repository.ipb.ac.id/handle/123456789/65199

[8] A.Z. Arifin, I.P.A.K. Mahendra, and H.T. Ciptaningtyas, "Enhanced Confix Stripping Stemmer and Ants Algorithm for Classifying News Document in Indonesian Language,", Proceeding of International Conference on Information & Communication Technology and Systems (ICTS), p. 149-157, 2009

[9] A. D. Tahitoe, and D. Purwitasari, "Implementasi Modifikasi Enhanced Confix Stripping Stemmer Untuk Bahasa Indonesia dengan Metode Corpus Based Stemming," Institut Teknologi Sepuluh Nopember (ITS) – Surabaya, 60111, Indonesia, 2010, http://digilib.its.ac.id/public/ITS-Undergraduate-14255-paperpdf.pdf

[10] D.N. Chandra, G. Indrawan, & I.N. Sukajaya, "Klasifikasi Berita Lokal Radar Malang Menggunakan Metode Naïve Bayes Dengan Fitur N-Gram, " Jurnal Ilmiah Teknologi Dan Informasi ASIA (JITIKA), 10(1),p.11–19. 2016 http://lp3m.asia.ac.id/wpcontent/uploads/2016/02/2.-jurnal-Denny.pdf

[11] E. Indrayuni, M. Wahyudi, S. Informasi, J. Selatan, I. Komputer, & J. Selatan, "Penerapan Charachter N-Gram Untuk Sentiment Review Hotel Menggunakan Algoritma Naive Bayes, " *Konfrensi Nasional Ilmu Pengetahuan dan Teknologi (KNIT)* (pp. 88–93). 2015.

[12] G. Dwyer, S. Aggarwal, J. Stouffer, "Flask: Building Python Web Services. Packt, ", Publishing Ltd, Birmingham, 2017

[13] Haroon, Danish and Karachi, "Python Machine Learning Case Studies, " Penerbit Apress, Pakistan, 2017

[14] R. Lawson, " Web Scraping with Python. Packt," Publishing Ltd, Birmingham, 2015

[15] MongoDB Compass. 2016. Dari : https://www.mongodb.com/blog/post/getting-startedwith-mongodb-compass. Diakses pada 25 Mei 2018 pukul 20.30 WIB.

[16] Gartner, "Preparing and Architecting for machine learning," 2017. https://www.gartner.com/binaries/content/assets/events/keywords/catalyst/catus8/preparing_and_architecting_for_machine_learning.pdf .